Report

On

Software Process Models

Submitted to:

Sir Muzaffar Hameed

Prepared by:

Muhammad Wasif Laeeq

BSIT0701

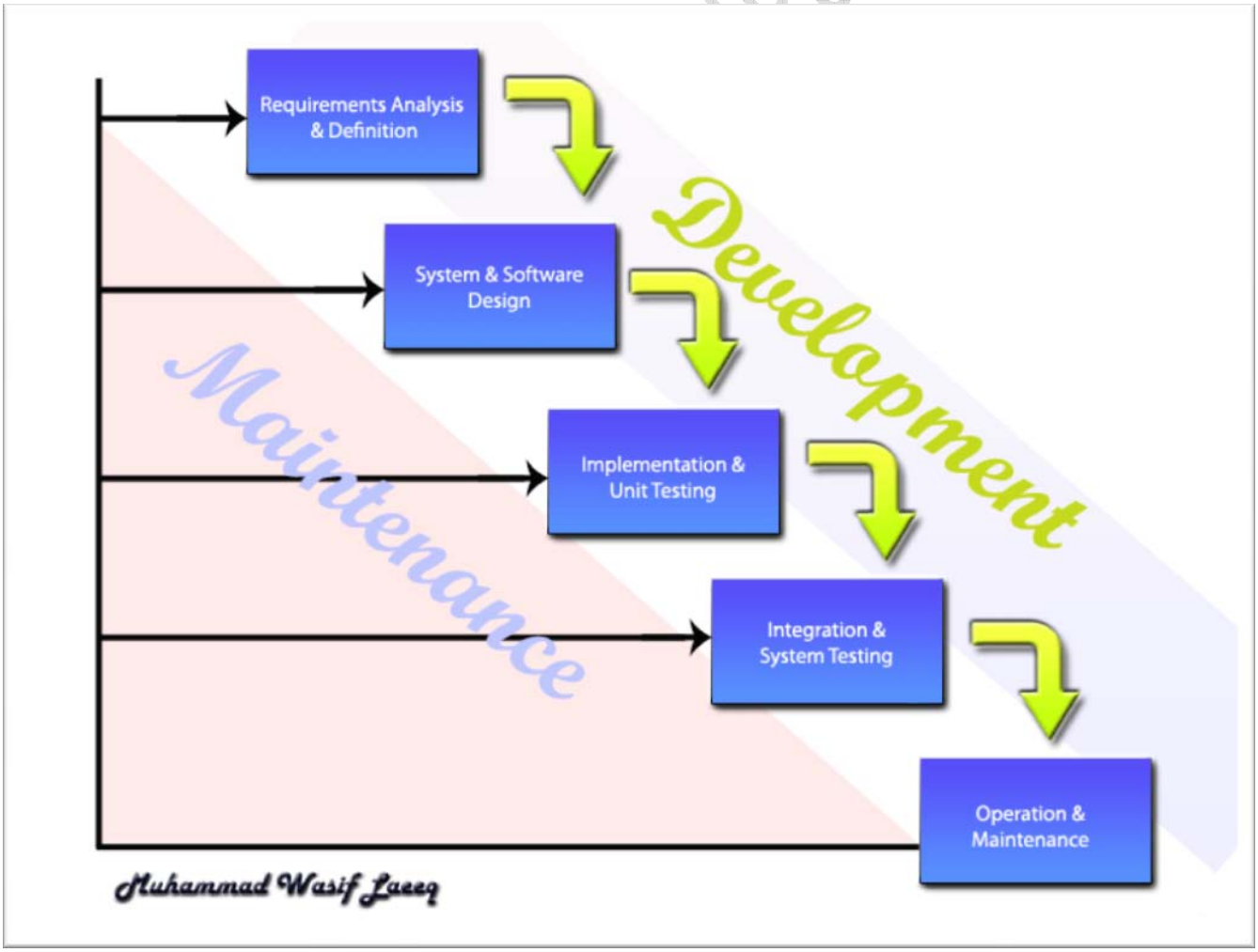
Department of Computer Science

Bahauddin Zakariya University Multan.

Contents:

1- Water Fall Model:

This model got its name due to cascading effect from one step to another as shown the figure below. It is the First Published model of software development process. This model is based on the article that was published by Royce in 1970. This model is sometimes referred as “Linear Sequential Model” or “Software Life Cycle”.



1.1- Watefall Model Phases:

It has following phases/steps;

1. Requirements analysis and definition
2. System & Software Design
3. Implementation & Unit Testing
4. Integration & System Testing
5. Operation & Maintenance

In Water fall model One Phase must be completed before stepping into the next phase, In each phase one or more documents are prepared and are approved (signed). If the Requirements phase is not completed we can't go forward for Design phase and so on. These phases overlap & provide information to each other. During Design phase problems with requirements are identified and during coding problems with the design are found and so on.

1- Requirement analysis & Definition:

In this phase, software development/engineering Company asks the user about the nature of project means, what the required project should do. Company will get the raw requirements (because user may just have the concept of software) from the user and will organize these requirements and analyses it. These requirements are then defined in detail and serve as a system specification. Constraints are also identified in this phase

2- System & Software Design:

In this phase, requirements (specifications) are implemented as a Graphical model using UML (Unified Modeling Language). Software design includes identifying and describing the fundamental software system abstractions and their relationships.

Along with software design, Hardware architecture is also considered. And software engineers determine that whether this software design will be able to be implemented on that specific (as described in requirement) Hardware architecture.

3- Implementation and Unit Testing:

In this phase, Different parts (modules) of the software specified in software design phase are converted into software, means the design is basically coded in a particular/specified language. These parts are called Units. Along with Development of the units, Units are also tested, and it is confirmed that all the units are working fine as the requirements were. Because we can't proceed further If a phase is not completed, so all the required units are developed at the End of this phase.

4- Integration & System Testing:

The Individual Units are integrated into One Single Software in this phase. And it is made possible that all the units work together in the same direction, means all modules should do their specified work even after integration with other units. If all units were working fine separately, But after integration our System is not working and is not giving a positive feedback then it will be useless.

So programmers must make it possible to work all units as a System. After the testing, software is given to user/customer.

5- Operation & Maintenance:

In this Phase, System is implemented at the customer's end, and customer starts using the software provided. But sometimes there may be Bug that was not discovered during the previous phases, so these bugs are to be removed and software is sent back to the respective phase. As specified in the Image, we can send the software for maintenance to any phase of this model.

OR there may come a new requirement for the customer, which he thinks that will create an ease for him. Then also we will do the maintenance. This makes this phase Longest Life-cycle phase. This is not a necessary phase.

1.2- Advantages of waterfall model:

The main advantage of waterfall model is that documentation is produced at every phase, and that documentations can collectively make the final documentation.

1.3- Disadvantages of Waterfall Model:

Disadvantage of this model is that, this model does inflexible partitioning of project. And if any new requirement comes from the customer, say at last phase, then we again have to go back to Requirements phase, define requirements again, design it again, change old units and/or develop new, integrate them again and then we will be able to continue.

Customer only sees the working software after it is completed; this may cause confusion if any problem is found

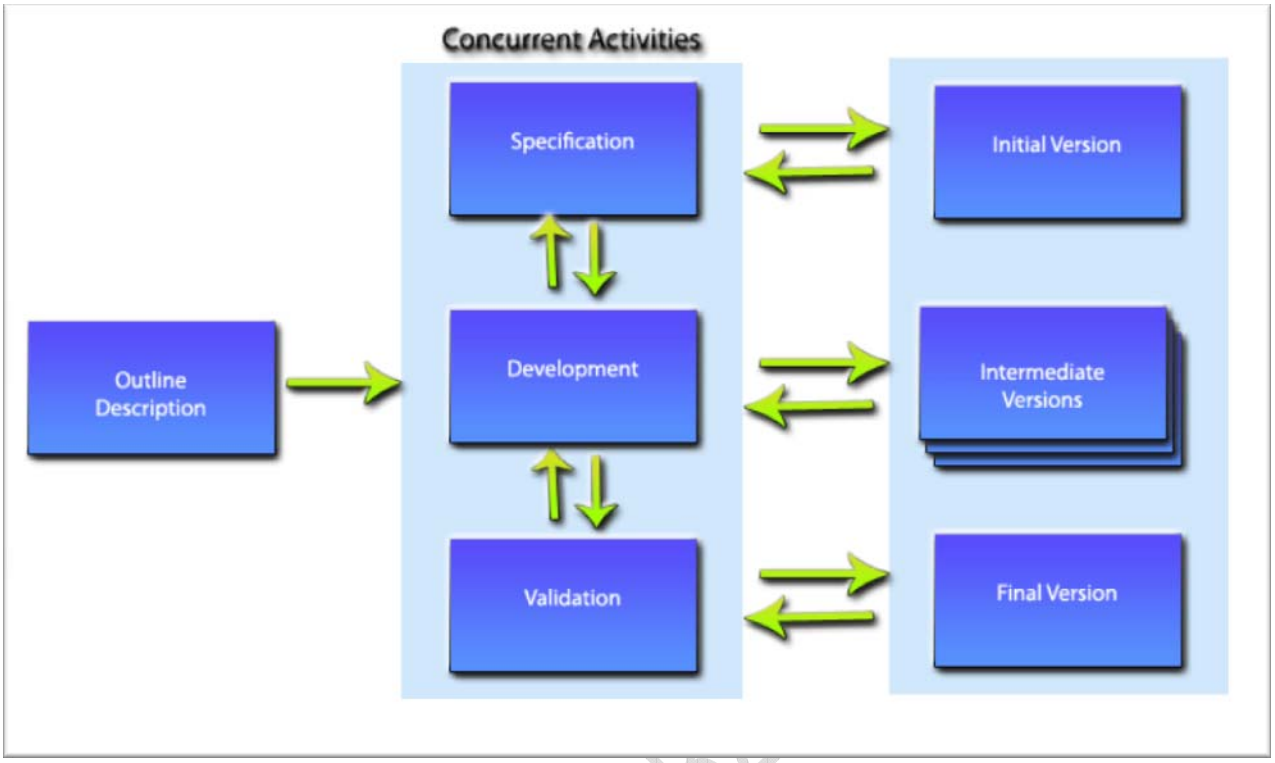
1.4- Conclusion:

Consider if the change in requirement occurs again & again then what happens, we will repeat the whole process again & again. It will increase the cost of software. And will also create difficulties to us.

So, this model is used where we have understood the requirements completely and we are almost sure that no further changes in the requirements will occur. Few business systems also has stable requirements e.g. sale/purchase system etc. This model is also used for large system engineering projects.

2- Evolutionary Development:

In evolutionary Development the main point is that, Customer is also kept involved in the development process. The idea which development company gets from customer, uses this idea and makes a prototype (Not Original Software, but shows some of requirements) exposes it to customer and gets more refined requirements from the customer. Refines the requirements and after several versions Final version is produced. In this model Specification, Development and Validation activities are interleaved with rapid feedback. This model is also called evolutionary prototyping because several prototypes are produced and each new prototype has evolved features than the previous one.



2.1- Types of Evolutionary Development

1. Exploratory Development
2. Throwaway Prototyping

1- Exploratory Development:

In this type, the development is started from the requirements which are understood to us. And we produce an initial version that fulfills the requirements which we understood. And we show this prototype to the customer, he sees the prototype and then asks to add some other features, requirements. Exploratory development is used to work with customer, to explore requirements and development of final version.

2- Throwaway Prototyping:

In this type, purpose of evolutionary development is to understand the requirements of customer, And to develop better requirements definition for the system.

Company creates a prototype based on vague requirements and shows to customer. Customer checks it and explains his requirements again. We develop another new prototype based on these new requirements. And by this process requirements are refined.

2.2- Problem in evolutionary Development:

1- Development Process is not visible. And it is not easy to develop documentation for each prototype as they are created so rapidly that if we create documentation for each build it will not be cost effective.

2- Due to continual change in the software coding and/or structure of software becomes so poor that it becomes difficult and costly for the evolution of system.

3- Component-based software engineering:

This is reuse-oriented approach of software development. In this model System is created by using existing COTS (Commercial off the shelf) Systems. This model is increasingly used as component standards are emerging.

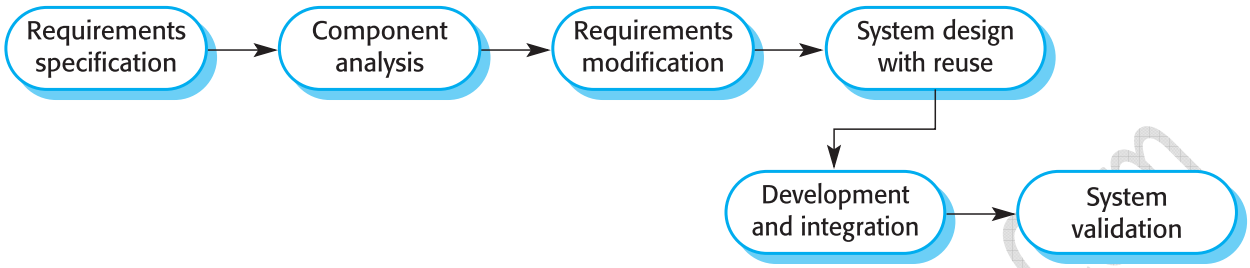
3.1- Process Stages:

1- ***Component Analysis***

2- ***Requirement modification***

3- ***System design with reuse***

4- **Development and Integration**

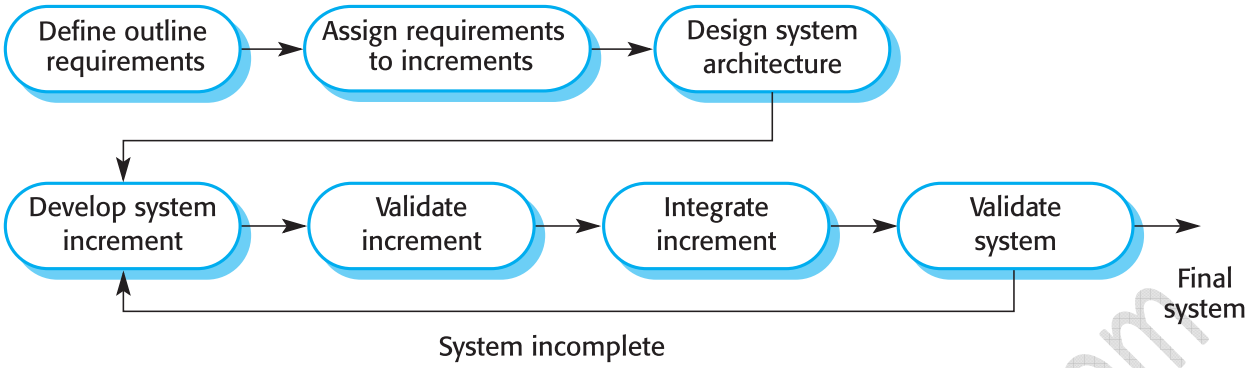


4- Incremental Development model:

In this model, Development is broken down into different parts/increments. Each increment fulfils a part of required functionality.

These increments are prioritized, and the one which has high priority is developed first and so on. And after each increment system is validated.

Once the development of an increment is started we will not add other requirements to this increment, any new requirements are added to the coming increment.



As shown in the figure above, If the system is still incomplete after increment validation , we will come back to development phase for the next increments, and if system is completed, we will release the final system

4.1- Advantages of Incremental Development:

1- Customer is attracted as he sees the early increments with some of his requirements working

2- With each increment requirements are evolved, and each increment works as prototype as was in evolutionary development.

3- There is low risk of overall project, because some of our increments are working fine. Initial increments have the high prioritized requirements. So high priority problem is solved at early stage means, most of the work is done.

4- As requirements that have highest priority are included in early increment, so as our project continues, we will test our project after each increment, So high prioritized services receive most of testing.

5- Spiral Development:

This development model is similar as a spiral, and each turn in the spiral represents a phase.

This model has no fixed phases; Phases are chosen as depending on the requirements.

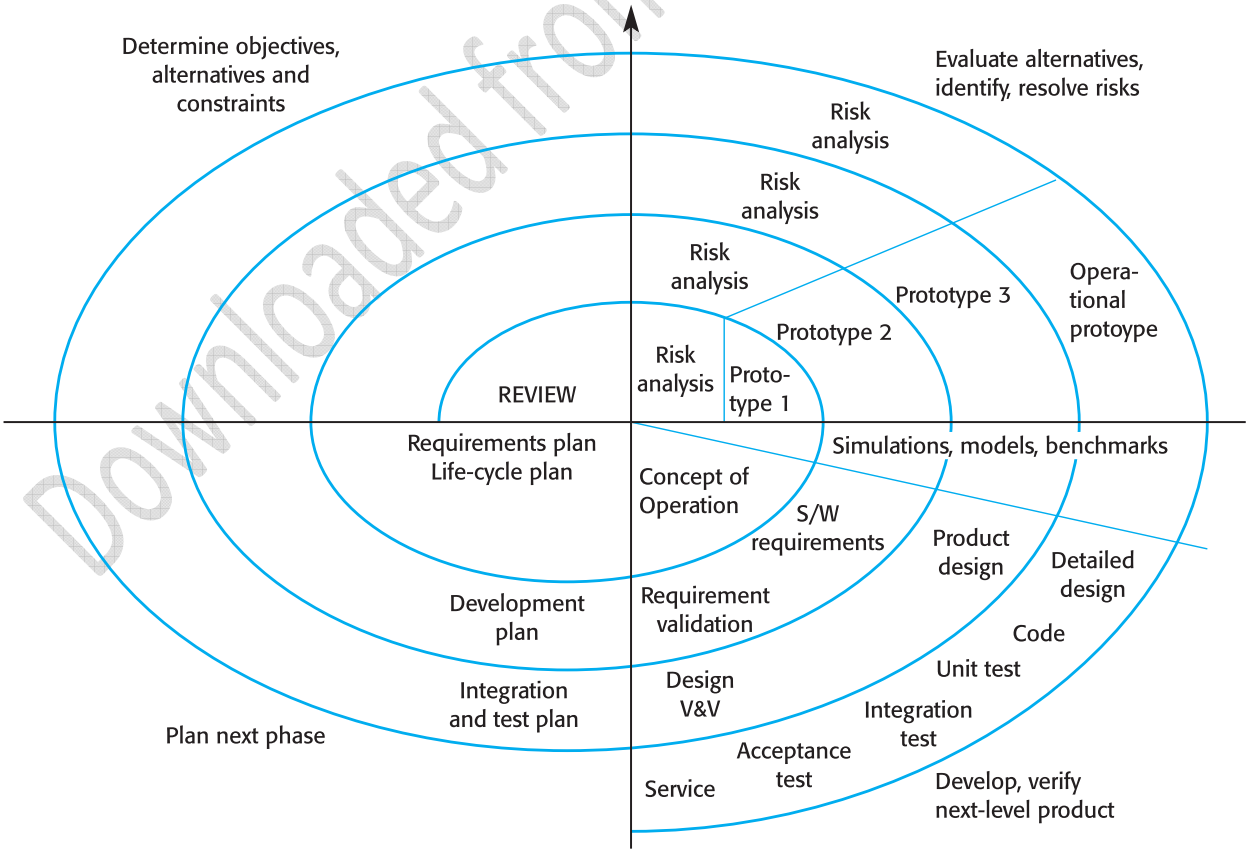
5.1- Spiral Model Sectors:

1- Objective Setting

2- Risk assessment & Reduction

3- Development & Validation

4- Planning



1- Objective Setting:

In this part of the loop, Objectives of phase are identified, means that at the end of this loop, our project will be able to do this particular job.

2- Risk assessment & Reduction:

Risks are assessed and activities put in place to reduce they key risks

3- Development & Validation:

In this part a development model is chosen, which can be anyone of the generic models. It can be Waterfall model, evolutionary model or Component based development. And the further development is done based on this chosen model.

4- Planning:

In this part the project is reviewed and it is planned that what we have to do in the next loop of spiral.

6- The Rational Unified Process (RUP)

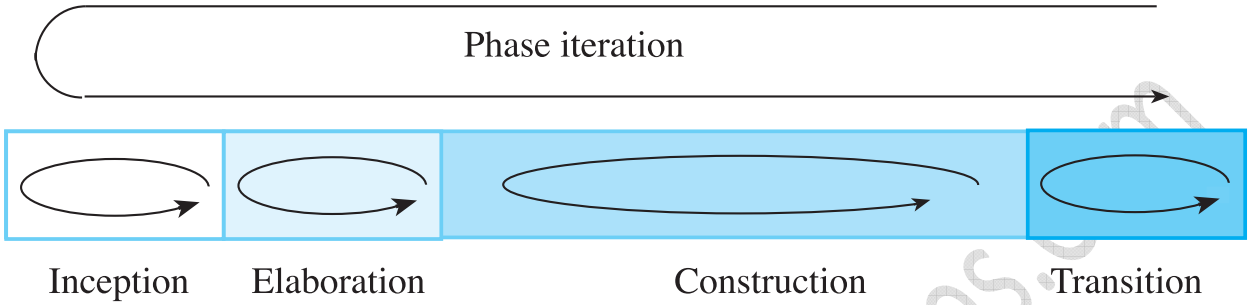
It is a modern process model, which is derived from the UML and other associated processes

RUP is normally described from 3 perspectives

1- A dynamic perspective that shows phases over time;

2- A static perspective that shows process activities;

3- A practice perspective that suggests good practice;



6.1- RUP Phases

1- Inception

2- Elaboration

3- Construction

4- Transition

Inception phase:

In this phase, the main objective is to scope the system adequately as a basis for validating initial costing and budgets. Business case, business context, success factors and financial forecast is established.

Elaboration Phase:

The elaboration phase is where the project starts to take shape. In this phase the problem domain analysis is made and the architecture of the project gets its basic form.

Construction Phase:

In this phase, the main objective is to design system, develop it and test the system. For larger projects, several construction iterations may be developed.

Transition Phase:

In this phase, the produced software is deployed in its working environment, and is made available for the use of end user.

6.2- RUP Good Practices:

Six fundamental best practices are recommended.

1- Develop Software Iteratively
2- Manage Requirements
3- Use Component Based architecture
4- Visually model software
5- Verify software quality
6- Control changes to software

1- Develop Software Iteratively:

Similar to incremental development, we should prioritize system feature and deliver high priority features in early builds.

2- Manage Requirements:

We should always consider the requirements, while accepting new requirements we should compare with the previous system and should see the effect of these requirements on old system. And for this whole tracking of requirements we have to always document the requirements.

3- Use component based architecture:

We should use component based development.

4- Visually model software:

We should use graphical representation of our software using UML. There are many softwares in market which provide easy modeling using UML. E.g.: IBM Rational Rose, Visual Paradigm for UML etc.

5- Verify Software Quality:

Ensure the quality of our produced software.

6- Control changes to software:

We must manage changes to software using a change management system and configuration management procedures and tools